BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

A General Solution Framework for Component-Commonality Problems

Nils Boysen, Department of Operations Management, Friedrich-Schiller-University Jena, E-mail: nils.boysen@uni-jena.de
Armin Scholl, Department of Decision Analysis, Friedrich-Schiller-University Jena, E-mail: armin.scholl@wiwi.uni-jena.de

Abstract
Component commonality - the use of the same version of a component across multiple products - is being increasingly considered as a promising way to offer high external variety while retaining low internal variety in operations. However, increasing commonality has both positive and negative cost effects, so that optimization approaches are required to identify an optimal commonality level. As components influence to a greater or lesser extent nearly every process step along the supply chain, it is not surprising that a multitude of diverging commonality problems is being investigated in literature, each of which are developing a specific algorithm designed for the respective commonality problem being considered. The paper on hand aims at a general framework which is flexible and efficient enough to be applied to a wide range of commonality problems. Such a procedure based on a two-stage graph approach is presented and tested. Finally, flexibility of the procedure is shown by customizing the framework to account for different types of commonality problems.

Keywords: Product variety, component-commonality, optimization, graph approach

Manuscript received September 24, 2008, accepted by Karl Inderfurth (Operations and Information Systems) March 26, 2009.

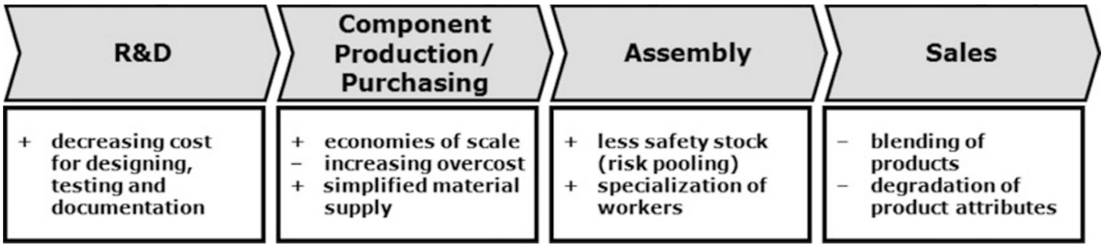
1 Introduction

In recent years, firms have more and more faced the necessity of providing an enlarged product variety, which nowadays seems inevitable in order to successfully serve highly diversified customer demands. For instance, some car series, especially from the luxury segment have billions of different car models (e.g., Boysen, Fliedner, and Scholl 2009b). In view of such an enormous variety, component-commonality - the use of the same version of a component across multiple products (Fisher, Ramdas, and Ulrich 1999) - is increasingly considered as a promising way to offer high external variety while retaining low internal variety in operations, and thus to lower cost (e.g., Swaminathan 2001). In this context, a firm has to solve the basic decision problem of how many and what kinds of components to utilize. The degrees of freedom for such a component-commonality problem range from providing a unique component for each single product up to a single component shared by all products (and any other solution in-between both extremes).

The extent of component-commonality influences (nearly) any process step along the supply chain (see Figure 1). In R&D, any additional component needs to be designed, tested and documented, and thus increases cost (e.g., Fisher, Ramdas, and Ulrich 1999). Moreover, if commonality is increased and fewer components in a larger quantity are to be produced (or purchased), economies of scale can be realized (e.g., due to a reduced number of setups and orders Tallon 1989 as well as intensified learning (Thonemann and Brandeau 2000)) and material supply to the final assembly are facilitated (Boysen, Fliedner, and Scholl 2009a). On the other hand, if multiple products share a common component, this component must meet specifications of the most demanding product, so that less discerning products receive a more valuable component than required (so-called overcost, see Briant and Naddef 2004). To decouple component production and final assembly, safety stocks need to be held, which can be reduced in size in cases of increasing commonality due to risk pool-

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Figure 1: Impact of increasing component-commonality along the supply chain



ing (e.g., Collier 1982; Baker, Magazine, and Nuttle 1986 . During final assembly, fewer components reduce the variability of operations for the workforce (Perera, Nagarur, and Tabucanon 1999). Finally in sales, commonality of visible components results in a blending of products, so that products become more indistinguishable from one another (e.g., Fisher, Ramdas, and Ulrich 1999). However, there is also a threat of negative impact from invisible components (e.g., a car battery) because indirectly product attributes might be degraded (e.g., increasing fuel consumption), see Ulrich (1995). Figure 1 summarizes the aforementioned effects of component-commonality along the supply chain, where positive and negative consequences of an increasing commonality are marked by '+' and '-', respectively.
Figure 1 depicts just a brief excerpt of the cost effects of common parts discussed in literature. More exhaustive reviews are provided, e.g., by Ramdas (2003), Swaminathan and Lee (2004) as well as Labro (2004). With regard to the variety of different relationships between component-commonality and supply chain operations it is not surprising that a massive body of literature has accumulated. Three major streams of research can be identified (see Labro 2004): (i) inventory- and operations-related commonality research, (ii) R&D- and engineering-related commonality research and (iii) marketing-related commonality research. Any of these streams covers a specific extract of the overall problem and any stream by itself contains a multitude of different research papers investigating specific component-commonality problems. Consequently, plenty of different solution approaches have been introduced, which are dedicated to the respective commonality problem being dealt with. The paper on hand aims at a general framework for solving component-commonality problems, which is both efficient and flexible enough to cover a multitude of different settings. For this purpose a two-stage graph approach is introduced, which can easily be customized for a specific commonality problem by simply changing the function to calculate arc weights in the graph.
The remainder of the paper is organized as follows: First, Section 2 provides a literature review on component-commonality. Then, Section 3 identifies a general core problem of component-commonality, which is formalized by a mathematical program. The solution framework is presented in Section 4 and initially described in solving the core problem of Section 3. Solution performance of this setting is tested in a comprehensive computational study in Section 5. Then, Section 6 shows how the solution framework can be adopted to cover extended versions of commonality problems taken from literature. Finally, Section 7 concludes the paper.

2 Literature review

As was already mentioned above, literature on component-commonality can be separated into three streams of research (see Labro 2004):

Inventory- and operations-related research: Dating back to the 1980s, component-commonality was initially investigated with regard to its influence on inventories and operations. A multitude of different models, e.g., provided by Collier (1982), McClain, Maxwell, Muckstadt, Thomas, and Weiss (1984), Baker (1985), Baker, Magazine, and Nuttle (1986), Gerchak, Magazine, and Gamble (1988), Eynan and Rosenblatt (1996), Thonemann and Brandeau (2000), Hillier (2000, 2002) and Ma, Wang, and Liu (2002), consider the benefits of common parts, which are, for instance, a decrease in order/setup and inventory cost due to the risk-pooling effect. On the one hand, setup costs are lowered by reducing the number of components,

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

as larger demands allow for larger lot sizes. On the other hand, having fewer components reduces the risk of forecast errors so that safety stock of such a multi-use component need not be as large as the sum of safety stocks of the covered specialized components. This effect is referred to as risk pooling, because additional demand for one product and reduced demand for another one using the same component might compensate each other.

R&D- and engineering-related research: Later on, commonality research more and more shifted focus from inventory and operations aspects to R&D and engineering considerations. One argument might be that the majority of operation cost is already determined during the engineering phase (e.g., Swift, Booker, and Edmondson 2004) and another that commonality is especially employed in a make-to-order environment where inventory aspects are negligible (see Jans, Degreave, and Schepens 2008). Nevertheless, some models intermix engineering- and inventory-related aspects (e.g., Dogramaci 1979; Thomas 1991). R&D especially benefits from common parts by avoiding duplicate development cost (Fisher, Ramdas, and Ulrich 1999; Perera, Nagarur, and Tabucanon 1999). Dogramaci (1979), Krishnan, Singh, and Tirupati (1999), Ramdas and Sawhney (2001) and Ramdas, Fisher, and Ulrich (2003) provide models for commonality problems where fixed costs for component development are a major element of the total cost function. Engineering-related commonality research typically restricts its models to the subset of components, which remain invisible to the customers (i.e., braking systems, Fisher, Ramdas, and Ulrich 1999, and Ramdas, Fisher, and Ulrich 2003, or wiring harnesses, Thonemann and Brandeau 2000).

Marketing-related research: Finally, if components visible to the customer are standardized, commonality has also an influence on sales, i.e., customer preferences are met less precisely, which, ultimately, decreases revenue. Research on this aspect of commonality stems, e.g., from Kim and Chhajed (2000), Desai, Kekre, Radhakrishnan, and Srinivasan (2001) as well as Heese and Swaminathan (2006). In an industry case presented by Jans, Degreave, and Schepens (2008), prices are calculated on a cost-plus basis, so that cost consequences of component-commonality indirectly influence revenues via the products' price elasticity. A recent approach considering sales and logistics aspects has been made by Ervolina, Ettl, Lee, and Peters (2009).

The so-called *assortment problem*, which has a long tradition stretching back more than five decades (see Hanssmann 1957; Sadowski 1959), can be seen as a forerunner of commonality research. For an extensive review on this problem see Pentico (2008). The assortment problem considers downward substitutability of products with just a single (significant) feature. As cost components, overcost and fixed cost for component development are to be minimized. Although the assortment problem was initially dedicated to stocking situations, it can be applied to a wide range of related situations, one of which is component-commonality. However, merely simple cost structures and just a single feature are considered, which hinders a direct application of the assortment problem in real-world commonality problems. The relationship of both fields of research is discussed in detail by Pentico (2008).

From a methodological point of view, commonality research mainly utilizes analytical models (e.g., Collier 1982; McClain, Maxwell, Muckstadt, Thomas, and Weiss 1984; Baker, Magazine, and Nuttle 1986; Gerchak, Magazine, and Gamble 1988; Desai, Kekre, Radhakrishnan, and Srinivasan 2001; Hillier, 2000, 2002; Ma, Wang, and Liu 2002) to gain general insights; nevertheless, also a wide arsenal of algorithmic optimization approaches is applied in literature to act as decision support in determining an optimal level of commonality. Plenty of exact procedures have been developed, i.e., mathematical programming (Briant and Naddef 2004; Jans, Degreave, and Schepens 2008), dynamic programming (Sadowski 1959; Rutenberg 1971), branch&bound (Thonemann and Brandeau 2000). Furthermore, a lot of heuristic approaches have been introduced in literature, i.e., clustering methods (Dogramaci 1979; Thomas 1991), priority rules (Gupta and Krishnan 1999), simulated annealing (Thonemann and Brandeau 2000), and decomposition approaches (Avella, Boccia, Martino, Oliviero, Sforza, and Vasil'ev 2005). All of these procedures were designed to cover a specific component-commonality problem, whereas our solution framework is flexible and efficient enough to be applied to a wide range of commonality problems. Moreover, our framework is able

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

to act both as an exact and a heuristic solution procedure.

3 A basic component-commonality problem

In this section, a basic component-commonality problem is developed, which exemplifies the elementary trade-off and exhibits all basic properties of more general component-commonality problems. By means of this basic problem version, the general course of our solution framework is described and solution performance is tested in Sections 4 and 5, respectively.

3.1 Problem description

A given set P of products with a given demand $d_p \forall p \in P$ is to be provided with components of a specific kind. Each product p has *minimum requirements* to be fulfilled by its designated component. These requirements refer to a set F of features owned by a component. Any feature $f \in F$ can receive different values $v \in V_f$, so that fixing a single value for each feature composes a complete specification of a component. Thus, a component-commonality problem has to answer three interrelated questions: (i) How many components with (ii) what specification to select and (iii) which product to provide with which component. To clarify our nomenclature the following example of automobile industry is given: Different car models (products) are to be supplied with sunroofs (component). A major property of sunroofs is the drive (feature), which might be manually (value 1) or electrically (value 2) powered.

Furthermore, it is assumed that values $v \in V_f$ per feature f are sorted in increasing order according to their ability to meet products' requirements, so that a value v per feature f is able to also fulfill a requirement for another value v' of the same feature, if $v' < v$ holds, but not vice versa. Literature on commonality labels this property as *downward compatibility* or *one-way substitutability*. For our example, this would mean that any customer would accept an electrical sunroof, if his/her minimum standard is a manual sunroof, but not the other way round. The minimum requirement value (some $v \in V_f$) of a product p with regard to feature f is denoted by the parameter r_{pf}.

In our basic commonality problem, we only consider two kinds of cost. On the one hand, *fixed cost* K occurs whenever an additional component is introduced and, e.g., represents all costs for developing, testing and documenting a component. On the other hand, *unit costs* of the components are captured, which originate from the realized specification of the respective component. As an elementary assumption, an additive cost structure is considered. So, the unit component costs are calculated by summing up the cost k_{fv} of the actually realized values v over all features f.

Table 1: Data of Example 1

	r_{pf}			
p	$f=1$	$f=2$	$f=3$	d_p
1	0	0	0	10
2	1	0	0	20
3	0	0	1	10
4	0	1	1	20
5	1	1	1	10
k_{f1}	1	1	1	K=20

Example 1: We consider $\mid P \mid = 5$ products with minimum requirements r_{pf} as given in Table 1. Each of the $\mid F \mid = 3$ features is present or not, so that $V_f = \{0, 1\} \forall f \in F$. In each case, the requirement is 0 (feature f is not needed by product p, i.e., $k_{f0} = 0$) or 1 (product p needs feature f). Thus, product 1 needs none of the features, while product 5 needs them all. Fixed cost K amounts to 20 monetary units. Any other data of our component-commonality problem is listed in Table 1, too. A possible solution for this example would be to introduce three components that serve products 1 and 2, 3 and 4, and product 5, respectively. The component for products 1 and 2 merely contains feature $f = 1$ and is to be produced $\sum_{p=1}^{2} d_p = 30$ times, so that variable and fixed costs amount to $k_{11} \cdot \sum_{p=1}^{2} d_p + K = 1 \cdot 30 + 20 = 50$. Since product 1 does not need the feature, overcost of $d_1 \cdot k_{11} = 10$ has to be paid for exceeding the requirement of product 1. This, however, spares paying the fixed component cost of $K = 20$. The overall cost D for the aforementioned solution results in $D = 180$, which is the optimal solution for this problem instance.

Example 2: Another instance with non-binary feature values is given in Table 2. It considers the configuration of battery types which are to be mounted into different car models. The three bat-

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 2: Data of Example 2

		r_{pf}		
p	$f=1$	$f=2$	$f=3$	d_p
1	1 (50 Ah)	1 (2 yrs)	1 (yes)	100
2	2 (70 Ah)	2 (4 yrs)	1 (yes)	50
3	3 (90 Ah)	2 (4 yrs)	2 (no)	30
4	3 (90 Ah)	3 (6 yrs)	1 (yes)	20
5	2 (70 Ah)	3 (6 yrs)	2 (no)	100
k_{f1}	10 (50 Ah)	15 (2 yrs)	5 (yes)	
k_{f2}	20 (70 Ah)	25 (4 yrs)	25 (no)	$K=2,500$
k_{f3}	30 (90Ah)	35 (6 yrs)	--	

tery features ($F=3$) are capacity ($f=1$, measured in ampere hours [Ah]), durability ($f=2$, measured in years), and maintenance ($f=3$, "yes" for required maintenance and "no" for maintenance-free). $P=5$ car models are considered which have the minimum requirements and demands specified in Table 2. Also given are the variable and fixed cost parameters. Note that downward compatibility allows that a car model p can get a battery which has the required or larger capacity, the required or longer durability and is maintenance-free even if $r_{p3}=yes$ is set. In the case of $r_{p3}=no$, only a maintenance-free battery is acceptable.
Two extreme solutions and the optimal solution are the following:

1. Each car model gets its own battery type and the fixed costs sum up to $5\cdot 2,500=12,500$. The unit cost of the battery type for product p results from the variable cost values $k_{f,r_{pf}}$ of the minimum requirements, i.e., car 1 gets a battery with 50 Ah, 2 years and maintenance: $k_{1r_{11}}+k_{2r_{12}}+k_{3r_{13}}=10+15+5=30$. The total variable cost is computed by multiplying unit cost with demands and summing up over all products resulting in $30\cdot 100+50\cdot 50+80\cdot 30+70\cdot 20+80\cdot 100=17,300$. Total cost is $12,500+17,300=29,800$.

2. The other extreme solution consists of a single component type which is mounted into any car model. In order to fulfill all requirements, this battery type must have the highest value for each feature (maximally equipped type) such that unit costs are $30+35+25=90$. The total cost amounts to $2,500+90\cdot 300=29,500$.

3. The optimal solution requires two battery types: The first type (70 Ah, 4 yrs, yes) with unit cost 50 is assigned to car models 1 and 2, while the second type (90 Ah, 6 yrs, no) with unit cost 90 is assigned to the models 3 to 5. Total cost amounts to $2\cdot 2,500+50\cdot 150+90\cdot 150=26,000$.

3.2 A nonlinear optimization model

In the decision model the specification of a component c is denoted by binary variables z_{cfv}, which receive value 1, whenever value v of feature f is realized in c (0, otherwise). The assignment of products $p\in P$ to components $c\in C$ is covered by binary variables x_{pc}, which are assigned value 1, if product p receives component c (0, otherwise). The binary variables y_c indicate whether a component c is actually chosen for production (1) or not (0). As the components are constructed within the model via the variables z_{cfv}, it is not necessary to enumerate all possible components (which would result in $\prod_{f\in F}|V_f|$ components). However, in order to restrict the number of variables to be defined in the model prior to computation, we use the simple insight that at most $|P|$ components are required in a solution. This maximal number would be obtained in the extreme case of doing without any component-commonality.
With the help of the notation summarized in Table 3 the basic core component-commonality-problem (CCCP) consists of the nonlinear objective function (2) and linear constraints (2) to (6):

$$(1)\quad \text{(CCCP) Minimize } D(X,Y,Z)=\sum_{c\in C}\sum_{f\in F}\sum_{v\in V_f} z_{cfv}\cdot k_{fv}\cdot\sum_{p\in P}x_{pc}\cdot d_p+\sum_{c\in C}y_c\cdot K$$

subject to

$$(2)\quad \sum_{c\in C}x_{pc}=1\ \forall p\in P$$

$$(3)\quad x_{pc}\le y_c\ \forall p\in P;\ c\in C$$

$$(4)\quad \sum_{v\in V_f}z_{cfv}=1\ \forall c\in C;\ f\in F$$

$$(5)\quad x_{pc}\cdot r_{pf}\le\sum_{v\in V_f}z_{cfv}\cdot v\quad \forall p\in P;\ c\in C;\ f\in F$$

$$(6)\quad y_c,x_{pc},z_{cfv}\in\{0,1\}\quad \forall p\in P;\ c\in C;\ f\in F;\ v\in V_f$$

In the objective function (2) total cost is to be minimized, which consists of variable cost (first term) and fixed cost (second term). Variable costs

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 3: Notation

P	set of products (index p)
C	set of components (index c)
F	set of features (index f)
V_f	set of values per feature f (index v)
k_{fv}	cost to realize value v of feature f per produced unit
K	fixed cost per component
r_{pf}	minimum requirement (some value $v \in V_f$) of product p with regard to feature f
d_p	demand for product p
y_c	binary variables: 1, component c is introduced; 0, otherwise
x_{pc}	binary variables: 1, product p receives component c; 0, otherwise
z_{cfv}	binary variables: 1, component c realizes value v of feature f; 0, otherwise

are calculated by multiplying unit cost per component, which is cumulated over the contained feature values, by the demand of those products that receive the respective component. Equations (2) ensure that each product receives exactly one component, whereas constraints (3) enforce that, if a component is assigned to a product ($x_{pc} = 1$) the component is to be introduced ($y_c = 1$), so that respective fixed costs accrue in the objective function.
Furthermore, it is to be ensured by inequalities (4) that each component must realize exactly one value per feature. Finally, constraints (5) enforce that minimum requirements of products are met. Whenever a component c is assigned to a product p ($x_{pc} = 1$) then the requirement r_{pf} of the product is to be satisfied by at least the same realized value or an even better one.

3.3 A linearized model

In order to solve the model by a standard MIP solver, it can be linearized by introducing additional binary variables that replace the nonlinear terms $x_{pc} \cdot z_{cfv}$ in the objective function:

$$q_{pcfv} = \begin{cases} 1, & \text{if component } c \text{ with value } v \text{ of feature } f \text{ is assigned to product } p \\ 0, & \text{otherwise} \end{cases}$$

The transformed model has the linear objective (7) and requires additional linear restrictions (8) to (11) which ensure $q_{pcfv} = x_{pc} \cdot z_{cfv} \quad \forall p, c, f, v$ without explicity demanding the binary property of the variables q_{pcfv}:

$$\text{(7)} \qquad \text{(CCCP') Minimize } D(X,Y,Z,Q) = \sum_{f \in F} \sum_{v \in V_f} k_{fv} \cdot \sum_{c \in C} \sum_{p \in P} q_{pcfv} \cdot d_p + \sum_{c \in C} y_c \cdot K$$

subject to (2) - (6) and
$\forall\, p \in P;\ c \in C;\ f \in F;\ v \in V_f$:

$$\begin{array}{llll} (8) & q_{pcfv} & \leq & x_{pc} \\ (9) & q_{pcfv} & \leq & z_{cfv} \\ (10) & q_{pcfv} & \geq & x_{pc} + z_{cfv} - 1 \\ (11) & q_{pcfv} & \geq & 0 \end{array}$$

Though quite possible, we do without reducing the sets of variables in order to keep the model comprehensible. When using modern MIP solvers, reduction routines are applied anyway. Furthermore, computational experiments indicate that no considerable gain in algorithmic performance can be obtained by (manual) variable reduction prior to starting the solver.

3.4 Related problems

The CCCP is NP-hard (see Briant 2000) and related to a number of other complex problems. For example, CCCP is related to the *uncapacitated facility location problem* (UFLP; e.g., Klose and Drexl 2005), where opening a facility represents introducing a component (connected with fixed cost K) and delivery costs are equivalent to total variable cost of component production. Downward compatibilities between products and their feature requirements r_{pf} can be transferred to a directed graph (see Briant and Naddef 2004). Each product receives a node in the graph (plus additional virtual products representing all additional combinations

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

of feature values). If a component designed to meet minimum requirements of a product i can also be integrated in a less demanding (real) product j, an arc (j, i) with arc weight $w_{ji} = \sum_{f \in F} k_{f,r_{if}} \cdot d_j$ is inserted. Any other arc required to build a complete graph is assigned an arc weight with a prohibitive large value.

With this graph on hand CCCP becomes equivalent to UFLP (which is also NP-hard, see Karup and Pruzan 1983). However, such a transformation requires to know beforehand the set of possible components, while in CCCP these components are designed by combining feature values (component design versus component selection). As the number of possible components increases exponentially in the number of features and values this transformation will usually not be useful in order to solve problem instances of real-world size.

A special case of CCCP with just a single feature can be solved in polynomial time, since the problem becomes an *assortment problem* with one-way substitutability (see Rutenberg 1971), which can be solved, e.g., with the famous Wagner-Whitin algorithm (Wagner and Whitin 1958) for dynamic single item *lotsizing* (see Sadowski 1959).

Referring to the classification of Pentico (2008), CCCP can be seen as an assortment problem with deterministic demand, a finite but potentially huge number of possible products/sizes, multiple product dimensions, a variable but limited number of products to stock/produce, a non-linear substitution cost structure and a stationary stocking pattern.

Furthermore, CCCP is related to lotsizing problems with product substitution (cf. Lang and Domschke 2008).

4 Solution framework

4.1 General procedure

The solution framework is based on a decomposition of the overall problem into two stages and resembles the solution procedure of Boysen and Fliedner (2008) for assembly line balancing which can be interpreted as a heuristic *metastrategy* for grouping problems:

- In the first stage, one or more different orders of products are determined by a suited heuristic method and stored in a set of sequence vectors.
- These product sequences are passed over to the second stage, where given orders of products are translated to a directed graph, which is applied to determine groups of products jointly served by a component and is, thus, labeled a *grouping graph*. Once a grouping graph is constructed, solving the component-commonality problem (for the set of given product sequences) reduces to finding the shortest path in the grouping graph.

The general idea of this solution framework is based on the following consideration. If products are ordered and stored in a sequence vector π, any possible grouping of products can be evaluated by a simple shortest-path-approach, provided that the following grouping policy is obeyed: Only products which are adjacent to each other in the product sequence π and, thus, form a subsequence of π may be unified to a product group. To allow for an intuitive understanding of this policy before the graph approach is formally described, Figure 2 displays a grouping graph for the given product order of $\pi =< 2, 3, 1]$.

As depicted in Figure 2 any possible grouping of products (represented by arcs and the sets of products stored with each arc) is contained in the graph, except for the product group $\{1, 2\}$, which would violate our grouping policy (products 1 and 2 are separated by product 3 within sequence π). If, furthermore, arc weights can be determined, which represent the cost associated with a component designed for the product set represented by the arc, then, solving the CCCP reduces to finding the shortest-path from source node 0 to the respective sink node. The length of the shortest-path equals the optimal total cost $D^*(\pi)$ for a given order π.

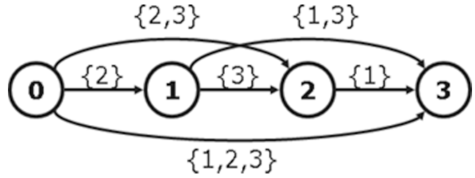
As an effective extension to the approach of Boysen and Fliedner (2008), this new method also constructs a grouping graph for multiple product sequences, so that, at the first stage, one or more promising orders of products need to be determined. A detailed and formal description of this drafted general idea is provided in the following subsections, where both stages are described in reverse order as this facilitates comprehension.

4.2 Stage 2: Grouping graph

Input of Stage 2 is a set Π of sequences $\pi_i \in \Pi$ with $i = 1, \ldots, |\Pi|$, each of which representing an order of products, so that products are stored

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Figure 2: Example grouping graph for a given product order of π=<2,3,1]



at sequence positions $\pi_i(s)$, with $s = 1, \ldots, |\ P\ |$. This input is applied to construct the grouping graph, which is defined as digraph $G = (V, E, c)$ composed of node set V, arc set E and an arc weighting function $c : E \to \mathbb{R}$, respectively.

The overall node set V is subdivided into stages $s = 1, \ldots, |\ P\ |$ plus an additional start node 0. Each stage s represents a sequence position and contains a subset $V_s \subseteq V$ of nodes. Stage s contains (up to) $|\ \Pi\ |$ nodes, one for each sequence. The ith node of stage s is denoted by $i(s)$ and represents the occurrence of products in the sequence π_i up to position s. The respective product set $P_{i(s)}$ is defined as follows: $P_{i(s)} = \{\pi_i(s') \mid s' = 1, ..., s\}$. Even different sequences might lead to identical subsets of products considered up to position s. To avoid additional computational effort for a duplicate inspection of identical nodes and associated product sets, only unique nodes $i(s)$ with regard to their product set $P_{i(s)}$ are generated:

$$(12) \quad V_s = \{i(s) \mid i = 1, ..., |\ \Pi\ |: P_{i(s)} \neq P_{j(s)}, \\ \forall j = 1, ..., i-1\}, \forall\, s = 1, ..., |\ P\ |$$

Note that avoiding duplicate product sets leads to a single node $1(|\ P\ |)$ in the final stage $s = |\ P\ |$, because any (feasible) order of products contains all products up to the final stage, so that:
$P_{i(|P|)} = P, \forall i = 1, \ldots, |\ \Pi\ |$.

The stage-dependent node sets V_s (plus initial start node 0) are unified to the overall node set V:

$$(13) \quad V = \bigcup_{s=1}^{|P|} V_s \cup \{0\}$$

After having defined the node set V, all nodes are renumbered consecutively from 0 to $n = |\ V\ | - 1$. Now, two nodes $i \in V_s$ and $j \in V_{s'}$ are connected by an arc, if the following conditions hold: (i) node j belongs to a later stage than node i, so that $s < s'$ holds and (ii) product set P_i of node i is a subset of product set P_j belonging to node j:

$$(14) \quad E = \{(i, j) \mid i \in V_s, j \in V_{s'}, \\ \text{with } s = 1, \ldots, |\ P\ | - 1,\ s' = s+1, \ldots, |\ P\ | \\ \text{and } P_i \subseteq P_j\}$$

Each arc represents a single component, which is dedicated to a special subset of products and, thus, has to fulfill all their requirements. This subset PS_{ij} of products assigned to an arc (i, j) is equal to the difference set, i.e., $PS_{ij} = P_j \setminus P_i$. Set PS_{ij} is stored with each arc and contains all products jointly served by the same component. This graph structure is a general element of the solution framework and remains unaltered irrespective of the specific component-commonality problem actually investigated.

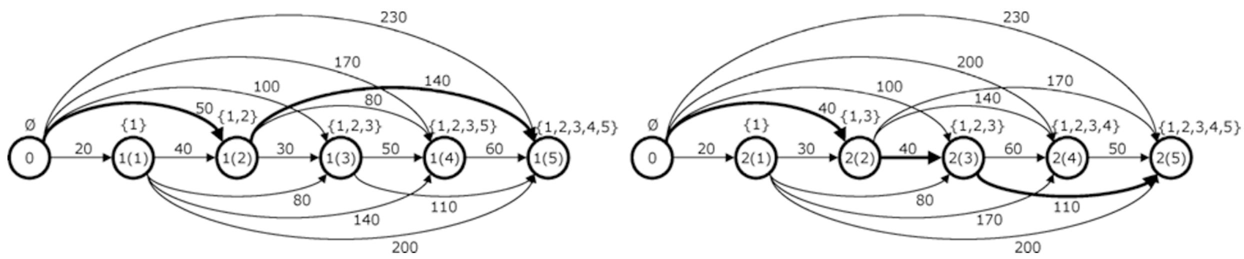
An arc weight represents the total cost of introducing the represented component. Consequently, its calculation depends on the specific cost structures of the respective commonality problem and is, thus, the basic element to customize our solution framework for a specific problem. In case of our basic commonality CCCP, an arc weight c_{ij} of an arc (i, j) receives variable cost VC_{ij} and fixed cost K: $c_{ij} = VC_{ij} + K\ \forall\, (i, j) \in E$, where variable cost VC_{ij} for $(i, j) \in E$ are calculated as follows:

$$(15) \quad VC_{ij} = \left(\sum_{f \in F} k_{fv_f^*}\right) \cdot \left(\sum_{p \in PS_{ij}} d_p\right) \\ \forall\, (i, j) \in E$$

The index $v_f^* = \max\{r_{pf} \mid p \in PS_{ij}\}$ denotes the value of the highest requirement per feature $f \in F$ of all assigned products from set PS_{ij}. With the help of index v_f^* the respective cost $k_{fv_f^*}$ per feature f can be identified, cumulated over all features and weighted with the overall demand of assigned products. How to adopt the calculation of arc weights to solve variational component-commonality problems is discussed in Section 6.

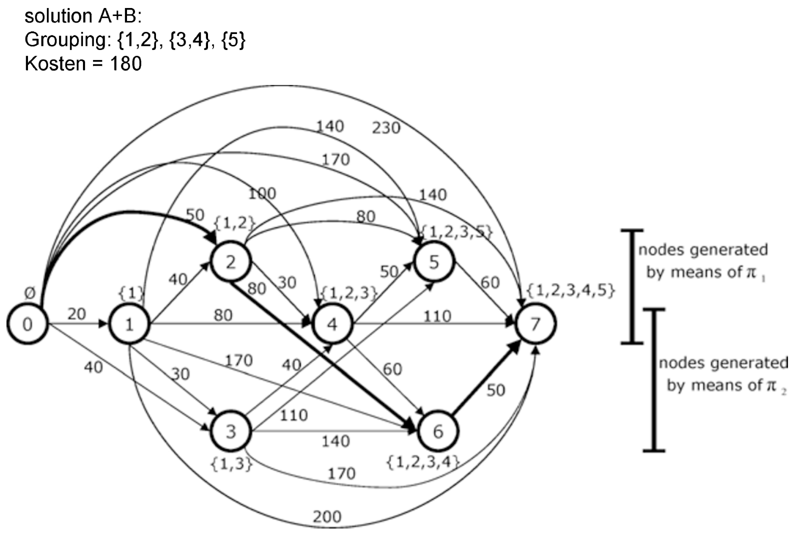
With such a grouping graph on hand, a component-commonality problem reduces to finding the shortest path from the unique source node 0 with product set $P_0 = \emptyset$ to the unique sink node n with an assigned product set of $P_n = P$. The length of the

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106



grouping graph for π_1={1,2,3,5,4}

grouping graph for π_2={1,3,2,4,5}



shortest path equals the minimum total cost $D^*(\Pi)$ for the given set of product sequences Π.

Note that the graph approach can also be applied if only a single product order ($|\Pi| = 1$) is determined at the first stage. However, as arcs allow for a cross over between different product sequences, the solution value D^* obtained by a unified grouping graph for a given set of product orders with $|\Pi| > 1$ is always better than or equal to a successive examination of isolated sequences $\pi_i \in \Pi$: $D^*(\Pi) \leq \min\{D^*(\pi_i) \mid i = 1, \ldots, |\Pi|\}$. This property is demonstrated by the following example.

Example: Given the problem instance of Table 1 and two product sequences $\pi_1 = \{1, 2, 3, 5, 4\}$ and $\pi_2 = \{1, 3, 2, 4, 5\}$. If a separate grouping graph is constructed for both sequences both solutions amount to an overall cost $D^*(\pi_1) = D^*(\pi_2) = 190$. The resulting two separated grouping graphs along with their bold-faced shortest paths are depicted in Figure 3. The grouping graph after unifying both single-sequence graphs and node renumbering is depicted in Figure 4. Former nodes $1(1)$ and $2(1)$ are merged to the new node 1, $1(2)$ and $2(2)$ get number 2 and 3, respectively. The nodes $1(3)$ and $2(3)$ are merged to new node 4 such that additional paths including the new shortest one become possible. The remaining nodes $1(4)$, $2(4)$, and the unique sink node $1(5) = 2(5)$ are given the new numbers 5 to $n = 7$. The shortest path through the combined grouping graph represents an improved solution to our CCCP instance with three components. The first is assigned to products 1 and 2, the second to products 3 and 4, and the third to product 5. Total cost (length of the shortest path) amounts to $D^*(\Pi) = 180$ which is, by chance, the minimal cost obtainable for our CCCP instance.

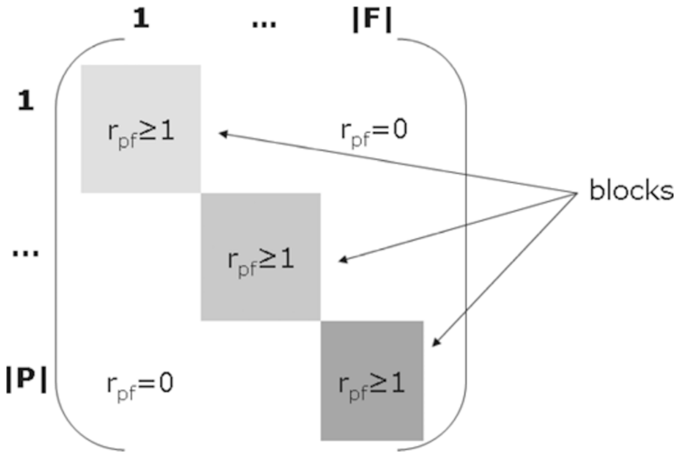
4.3 Stage 1: Sequencing of products

There exist numerous alternatives of how to determine adequate product sequences. These alternatives can, i.e., be classified by the number of product sequences generated:

- If *all* possible successions of products are gen-

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Figure 5: Intended block structure after binary sorting of requirements matrix r



erated and passed over to Stage 2, obviously the overall optimal solution with minimal cost D^* is determined and our approach acts as an exact solution procedure. However, such a complete enumeration suffers from the extraordinary number of possible sequences, which is $\frac{|P|!}{2}$. Thus, the ability of our solution framework to act as an exact solution procedure is more a theoretical one, especially if problem instances of real-world size are to be solved. Anyhow, this property is useful, if heuristic settings of our framework are to be evaluated according to their solution quality.

- On the other hand, only a *single* sequence can be produced. In this case, computational effort is reduced for the price of solution quality. One possibility would be to adopt a binary sorting procedure, which is, e.g., often applied to the so-called *cell-formation-problem* in Group Technology (see King and Nakornchai 1982; Burbidge 1991). This problem deals with forming groups of products, which are jointly produced in a separate shop and require similar resources to reduce investment cost.
  To adopt binary sorting, all features are to be resorted in ascending order according to the following priority value w_f, where v^* is the maximum number of different values per feature (including absence of the feature, i.e., value 0) over all products: $v^* = \max\{| V_f \; | | f \in F\}$:

  $$(16) \quad w_f = \sum_{p \in P} r_{pf} \cdot (v^*)^{(|P|-p)} \quad \forall f \in F$$

  Finally, the resulting reordered requirements matrix $\mathbf{r}$ is applied to determine an initial product sequence π according to the following priority value u_p in descending order:

  $$(17) \quad u_p = \sum_{f \in F} r_{pf} \cdot (v^*)^{(|F|-f)} \quad \forall p \in P$$

  This procedure resorts the requirements matrix $\mathbf{r}$, so that blocks of similar requirements can be identified (exemplified by Figure 5). According to this block structure Equations (17) assign each product p a priority value u_p.

  Example (cont.): The resulting product sequence is $\pi = \{5, 2, 4, 3, 1\}$. The optimal grouping for this sequence, $\{5\}, \{2\}, \{4\}, \{1, 3\}$, obtained by the grouping graph results in total cost of $D^*(\pi) = 190$.

- A compromise between both extremes would be to produce *some* solutions. A very simple advancement would be to approach a random sampling and to determine a number x of randomly drawn sequences. A more sophisticated approach to identify a promising subset of product sequences would be to apply a meta heuristic.
  In the following, an Ant Colony approach (see Dorigo, Caro, and Gambardella 1999) is developed. In an *Ant Colony approach*, solutions are constructed repetitively by software agents (artificial ants), which typically base their decisions on some local heuristic measure and the collected experiences of all former ants, aggregated in a so-called pheromone matrix. The search process of an individual ant resembles a simple priority rule-based heuristic, such that

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

at each sequence position s a single product is chosen out of the set POS_s of possible alternatives (products not yet scheduled). An ant's sequence π_i is hence filled from left to right. However, the choices of an ant are not deterministic, but stochastic according to a weighted probability scheme which is repetitively calculated at each decision point (sequencing position). The probability $Prob(p, s)$ that product p is assigned to position s is then determined on the basis of its priority value $w(p, p^{s-1})$ and the intensity of the pheromone $\tau_{pp^{s-1}}$ with respect to its alternatives, where p^{s-1} is the previously scheduled product in the sequence $p^{s-1} = \pi_i(s-1)$:

$$\text{(18)} \quad Prob(p,s) = (\tau_{pp^{s-1}})^{\alpha} \cdot \left(\frac{1}{w(p,p^{s-1})}\right)^{\beta} \times \frac{1}{\sum_{p' \in POS_s} (\tau_{p'p^{s-1}})^{\alpha} \cdot \left(\frac{1}{w(p',p^{s-1})}\right)^{\beta}} \quad \forall\, s = 2, \ldots, \mid P \mid;\, p \in POS_s$$

As priority value $w(p, p^{s-1})$ we simply measure the similarity between the previously scheduled product p^{s-1} and candidate product p according to priority value w_p of equation (17): $w(p, p^{s-1}) = \mid w_{p^{s-1}} - w_p \mid \ \forall\, p \in POS_s$. Analogously, pheromone value $\tau_{pp^{s-1}}$ is determined between predecessor p^{s-1} and actual product p, so that pheromone is stored in a $\mid P \mid \times \mid P \mid$-matrix. The initial product of each ant's sequence is randomly drawn. Parameters α and β control the relative importance of the pheromone versus the priority values. Because of experiences with other sequencing problems reported in the literature, these parameters are set to $\alpha = 1$ and $\beta = 2$ (see Stützle and Dorigo 1999).

In this way, all ants belonging to the actual iteration k construct their respective sequence. Once all $\mid \Pi \mid$ sequences are generated, this set of sequences is passed over to Stage 2, where the grouping graph is constructed and the best product grouping for the iteration is determined. Note that each stage's grouping graph is discarded after having determined the respective solution. This way computational effort for constructing additional arcs is restricted for the price of losing information about promising groupings. The optimal solution of iteration k can be retranslated into an optimal sequence $\pi(k)$, which along with the corresponding solution value $D^*(\pi(k))$ is applied to update the pheromone trail. Thus, pheromone value $\tau_{pp'}(k)$ in iteration k is calculated as follows:

$$\text{(19)} \quad \tau_{pp'}(k) = \tau_{pp'}(k-1) \cdot (1-\rho) + \rho \cdot \begin{cases} \frac{1}{D^*(\pi(k))}, \text{ if } p \text{ and} \\ p' \text{ are neighbors in } \pi(k) \\ 0, \text{ otherwise} \end{cases} \quad \forall\, p, p' \in P$$

The formula incorporates two mechanisms for guiding the search. Older pheromone is constantly reduced (evaporation) which strengthens the influence of more recent solutions and new pheromone is assigned to all product successions, which are part of the solution, in proportion to the respective objective value. The parameter ρ, which is set to 0.5, controls the relative importance of these two components. Note that the pheromone matrix has to be initialized with starting values $\tau_{pp'}(0) = \frac{1}{D^*(\pi_{start})} \forall\, p, p' \in P$, where π_{start} represents a first, randomly drawn product sequence. In the current implementation 20 ants are employed to construct solutions in any iteration. After 500 iterations the algorithm terminates and the best solution found is returned.

Which alternative of sequence generation is an appropriate choice mainly depends on the computational effort a planner is willing to spend. A more detailed answer can be stated with the help of the computational study in the following section.

5 Computational study

Up to now, commonality research exclusively investigates different special problem settings mostly inspired from real-world cases. Consequently, no established test bed for our basic commonality problem CCCP is available. Therefore, we first

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

elaborate on the instances that are used in our computational study. Then, experimental results on the performance of algorithms are presented.

5.1 Instance generation

In our computational study, we distinguish between two classes of test instances: small and large instances. The small instances are designed such that our solution framework can still solve all test instances to optimality (in acceptable time). Large instances shall represent problem instances of a size relevant in real-world settings, where only heuristic solutions are obtainable. To derive these instance classes the input parameters listed in Table 4 are used to produce the requirements of products r_{pf}, product demands d_p and variable cost k_{fv} per feature and value defining a CCCP-instance.
Within each test case, these parameters are combined in a full-factorial design and instance generation per parameter constellation is repeated 10 times, so that $2 \cdot 6 \cdot 5 \cdot 10 = 600$ different CCCP-instances are obtained. On the basis of a given set of parameters each single instance is generated as follows:

- *Product requirements:* First, the number of values V_f per feature f is randomly determined by drawing an uniformly distributed integer out of the interval $[1, V_f^{max}]$. Then, the products' requirements r_{pf} are fixed by randomly drawing an uniformly distributed integer out of the interval $[0, V_f]$ in each case.
- *Product demands:* The demands d_p of products p are randomly drawn with uniform distribution out of the interval $[1, 1000]$.
- *Variable cost:* Finally, a feature specific real value ϱ^f, which is the basic cost factor per feature f, is randomly drawn (with uniform distribution) out of interval $[0.5, 1.5]$. This factor is applied to determine variable cost k_{vf} per value v of feature f: $k_{fv} = \varrho^f \cdot v \; \forall \, f \in F; \, v = 1, \ldots, V_f$.

All generated instances can be downloaded from the online platform of this e-journal.

5.2 Performance of algorithms

All methods have been implemented in C# (Visual Studio 2003) and run on a Pentium IV, 1800 MHz PC, with 512 MB of memory. First, the performance of the procedures with regard to the small instances is evaluated (see Table 5). These instances can be solved to optimality by a complete enumeration (labeled ENUM) of all possible product sequences (only reverted sequences of already generated ones can be left out, see Section 4.3). For this exact procedure, we report the average solution time, measured in CPU-seconds and abbreviated by avg cpu. Compared to optimal objective values, solution performance for our solution framework is reported if the priority rule approach (PRIO), a random sampling (RAND) of 20 sequences and our Ant Colony approach (ANTS) is applied in the first stage, respectively. As a benchmark procedure, we apply standard MIP solver XPress-MP (optimizer version 18.10) to the linearized model of Section 3.3 which has been coded by means of XPress Mosel (version 2.2.0). Since some instances could not be solved to optimality even in cases of very long run times, the computing time was restricted to 500 seconds per instance. Thus, in time-out cases, XPress-MP only finds a heuristic solution.
To capture solution performance, Table 5 lists the average (maximum) relative deviation from the optimum (labeled avg gap (max gap)) in percentage for any parameter constellation, where deviations per instance are measured by:

$$\frac{D(x) - D(ENUM)}{D(ENUM)} \cdot 100\% \qquad \forall x \in \{\text{PRIO, RAND, ANTS, XPRESS}\}$$

Table 5 reveals an exponential increase of solution time required to determine an optimal solution by ENUM with increasing number $\mid P \mid$ of products. This result is not surprising because the number of sequences to be evaluated increases factorially in $\mid P \mid$, as well. On the other hand, solution times of PRIO and RAND are negligible as within neither instance more than 0.1 CPU-seconds are required. Both heuristic approaches, PRIO and RAND, show very promising results as the average gap over all instances amounts to merely 1.2% and 0.7%, respectively. According to the trade-off between solution time and quality, ANTS ranges in between. It solves a remarkable number of 273 instances (93%) to optimality with an average gap of merely 0.1% at an average computational time of 1.8 CPU-seconds.
This positive result is further underlined by the

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 4: Parameters for instance generation

symbol	description	values small	values large
$\mid P \mid$	number of products	$5, 6, \ldots, 10$	$75, 100, \ldots, 200$
$\mid F \mid$	number of features	$3, 4, \ldots, 7$	
V_f^{max}	maximum number of (non-zero) values per feature	4	
K	fixed cost for component development	5000	

worse performance of the off-the-shelf MIP solver XPress-MP. While it is able to solve small instances with 5 to 6 products in reasonable time, it fails in solving the larger instances to optimality regularly. ANTS clearly outperforms XPress-MP with respect to solution quality and time. Compared to ENUM, it becomes obvious that the sequence-based approach of our two-stage procedure is promising as a complete enumeration is much more efficient than solving the model with XPress-MP.
Table 6 lists the results for the large test instances. Here, optimal solutions remain unknown, so that the quality measures, avg gap and max gap, are calculated in relation to the best objective value obtained per instance by one of the three procedures, PRIO, RAND and ANTS. To reasonably restrict computational time, the number of iterations of ANTS is bound to 20 with 5 ants generating sequences per iteration, whereas RAND is executed in an unchanged setting generating 20 random product sequences.
Note that applying XPress-MP to the large instances turned out to be impossible as the solver even failed in solving the LP-relaxation of most of the instances within the time limit of 500 seconds such that not even a feasible solution could be found in these cases.
With regard to solution quality, ANTS shows superior. It contributes the best objective value to any instance except for three (where RAND finds the best solution). However, ANTS requires a considerable amount of computational time with an average of 115 CPU-seconds. With $\mid P \mid = 200$ products the maximum computational time is 363 CPU-seconds. Thus, instances with $\mid P \mid > 200$ can only be solved by ANTS if even more computational time is accepted or the number of iterations and ants is further reduced. Again, PRIO requires least computational time with an average of only 0.4 CPU-seconds. What is even more, the average gap amounts to merely 1.6%. Finally, RAND shows not competitive, as it is inferior with regard to both time and quality compared to PRIO. In instances of real-world size the solution space seems far too large, so that in contrast to the small instances a random sampling is not able to cover a sufficient proportion of all possible product sequences. Consequently, our priority rule based approach (PRIO) seems best suited for generating near optimal solutions, whenever instances of real-world size are to be solved in a very short time frame.

6 Customizing the solution framework

In this section, different extensions of our basic commonality problem CCCP are investigated for how to customize our solution framework. These extensions are subdivided according to the aforementioned classification of Labro (2004) into inventory-, engineering- and marketing-related issues of component-commonality.

6.1 Inventory and operations related extensions

Inventory and setup cost:
Whenever components are produced to stock, inventory cost accrue and a reorder policy needs to be applied. Inspired by a real-world commonality problem of wiring harnesses, Thonemann and Brandeau (2000) model a continuous review (R_c, Q_c) policy. That is, whenever the stored quantity of a component $c \in C$ reaches the reorder point R_c, a new order of quantity Q_c is placed. Delivery requires a constant lead time τ. Additionally, it is assumed that for each component a fill rate of β should be guaranteed (β-service level), i.e., $\beta \cdot 100\%$ of all orders have to be fulfilled directly from stock for all components.
The demands of products $p \in P$ are assumed to be independent random variables with expected demand rates d_p (average demand per period) and

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 5: Performance of procedures for small instances

		ENUM	PRIO		RAND		ANTS			XPRESS		
$\|P\|$	$\|F\|$	avg cpu	avg gap	max gap	avg gap	max gap	avg gap	max gap	avg cpu	avg gap	max gap	avg cpu
5	3	0.02	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.0	0.0	0.4
	4	0.02	0.8	5.7	0.0	0.1	0.0	0.0	0.4	0.0	0.0	1.3
	5	0.02	1.5	11.1	0.0	0.0	0.0	0.0	0.4	0.0	0.0	2.4
	6	0.02	1.2	5.9	0.0	0.0	0.0	0.0	0.5	0.0	0.0	3.6
	7	0.02	1.5	12.9	0.0	0.0	0.1	0.8	0.5	0.0	0.0	3.4
6	3	0.10	0.2	0.9	0.0	0.0	0.1	0.8	0.7	0.0	0.0	16.5
	4	0.09	0.9	4.7	0.0	0.0	0.0	0.0	0.7	0.0	0.0	22.1
	5	0.10	1.1	3.7	0.3	2.6	0.0	0.0	0.7	0.0	0.0	18.6
	6	0.10	0.7	5.0	0.2	1.7	0.0	0.0	0.8	0.0	0.0	33.6
	7	0.10	2.2	5.7	0.8	4.3	0.0	0.0	0.8	0.0	0.0	76.5
7	3	0.67	0.9	5.5	0.6	3.9	0.0	0.0	1.0	0.0	0.0	101.8
	4	0.68	0.1	0.8	0.3	2.6	0.0	0.0	1.1	0.0	0.0	143.5
	5	0.68	1.0	3.4	0.3	2.7	0.0	0.0	1.2	0.0	0.0	218.8
	6	0.67	0.6	1.9	0.9	4.5	0.0	0.0	1.3	0.0	0.0	165.7
	7	0.68	1.6	7.1	0.5	3.9	0.0	0.0	1.3	0.0	0.0	241.9
8	3	5.52	0.2	1.5	0.3	2.8	0.0	0.0	1.5	0.0	0.0	221.5
	4	5.46	1.2	4.8	0.7	3.8	0.0	0.0	1.7	0.0	0.0	419.7
	5	5.44	0.7	2.8	1.2	7.5	0.1	0.5	1.8	0.2	1.6	444.7
	6	5.45	1.4	5.0	1.1	4.4	0.2	1.5	1.9	0.0	0.0	431.4
	7	5.45	2.3	6.1	1.0	3.2	0.1	0.7	2.1	0.5	2.5	508.4
9	3	50.8	0.9	8.5	0.2	2.5	0.0	0.0	2.2	0.0	0.0	307.3
	4	51.0	0.4	2.0	0.5	2.7	0.0	0.3	2.4	0.1	1.2	458.5
	5	50.8	1.8	8.2	0.6	2.5	0.0	0.0	2.8	0.2	1.7	508.2
	6	50.7	1.5	4.8	0.7	3.4	0.2	1.8	3.0	0.6	2.8	508.2
	7	50.7	1.9	5.9	1.3	2.4	0.1	0.7	3.2	1.1	3.9	506.8
10	3	558.30	1.3	6.2	1.9	7.1	0.0	0.0	3.3	0.2	2.2	457.4
	4	540.60	0.9	3.1	2.0	4.8	0.1	1.4	3.6	0.0	0.0	463.8
	5	536.80	2.7	6.5	0.9	2.4	0.3	2.8	4.0	2.6	5.9	512.3
	6	542.50	2.7	6.7	2.4	4.2	0.2	1.2	4.1	1.7	4.1	507.1
	7	534.60	2.3	5.9	1.4	2.7	0.2	1.9	4.7	1.7	3.7	507.4
	all	99.90	1.2	12.9	0.7	7.5	0.1	2.8	1.8	0.3	5.9	260.4

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 6: Performance of procedures for large instances

		PRIO			RAND			ANTS		
$\|P\|$	$\|F\|$	avg gap	max gap	avg cpu	avg gap	max gap	avg cpu	avg gap	max gap	avg cpu
75	3	2.3	4.0	<0.1	10.9	18.0	4	0.0	0.0	11
	4	2.1	3.9	0.1	5.5	10.2	4	0.0	0.0	13
	5	2.4	4.0	0.1	6.2	10.2	5	0.0	0.0	15
	6	1.9	3.2	0.1	4.5	6.2	5	0.0	0.0	17
	7	1.9	2.6	0.1	3.6	4.6	6	0.0	0.0	19
100	3	1.5	3.2	0.1	8.5	15.4	7	0.0	0.0	25
	4	1.7	2.8	0.1	8.3	15.4	8	0.0	0.0	31
	5	1.6	3.4	0.1	5.5	9.6	9	0.0	0.0	35
	6	1.6	2.1	0.1	4.6	8.3	11	0.0	0.0	40
	7	2.2	3.7	0.2	5.0	9.4	12	0.0	0.0	45
125	3	2.0	4.9	0.2	12.6	22.7	12	0.0	0.0	52
	4	2.0	3.4	0.2	8.7	15.9	14	0.0	0.0	60
	5	1.7	3.9	0.2	7.3	11.6	16	0.0	0.0	68
	6	1.9	2.8	0.3	5.2	8.2	18	0.0	0.0	77
	7	1.4	2.2	0.3	4.8	9.4	20	0.0	0.0	87
150	3	0.5	2.3	0.5	15.8	30.5	20	0.0	0.0	87
	4	1.2	3.0	0.5	8.9	16.1	23	0.0	0.0	103
	5	1.3	2.8	0.6	6.2	9.6	26	0.0	0.0	119
	6	1.3	1.9	0.7	6.0	9.8	29	0.0	0.0	133
	7	1.5	2.2	0.8	4.7	7.7	32	0.0	0.0	148
175	3	1.0	3.3	0.5	12.1	22.9	30	0.0	0.0	139
	4	1.7	3.0	0.5	8.7	11.3	33	0.0	0.0	161
	5	1.6	3.6	0.6	6.6	9.4	29	0.0	0.0	186
	6	1.6	2.8	0.7	4.7	6.7	41	0.0	0.0	206
	7	1.4	3.1	0.8	4.2	7.1	44	<0.1	0.2	228
200	3	0.6	2.6	0.7	14.1	19.4	40	0.0	0.0	201
	4	1.6	2.7	0.8	7.5	10.2	45	<0.1	0.3	233
	5	1.4	2.8	0.9	6.2	8.9	51	0.0	0.0	257
	6	1.3	2.1	1.0	5.6	8.8	57	0.0	0.0	302
	7	1.4	2.1	1.2	4.6	6.0	64	0.0	0.0	339
	all	1.6	4.9	0.4	7.2	30.5	24	<0.1	0.3	115

standard deviations σ_p for the cumulated demand during the replenishment lead time τ. These parameters are used to define expected demand rates μ_{ij} and standard deviations σ_{ij} of lead-time demands for all components, which are represented by arcs $(i,j) \in E$ within our solution framework (see Section 4):

$$\mu_{ij} = \sum_{p \in PS_{ij}} d_p \quad \forall (i,j) \in E \tag{20}$$

(expected demand of components)

$$\sigma_{ij} = \sqrt{\sum_{p \in PS_{ij}} (\sigma_p)^2} \quad \forall (i,j) \in E \tag{21}$$

(std. dev. of demand in lead time)

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Unit inventory holding cost rates h_{ij} per time unit for any component are computed by multiplying unit cost (measuring the economic value added, i.e., the capital locked) with a constant interest rate $\bar{h}$ (see Equations (15)):

$$(22) \quad h_{ij} = \bar{h} \cdot \left(\sum_{f \in F} k_{fv_f^*} \right) \quad \forall (i,j) \in E$$

The order quantities Q_{ij} and reorder points R_{ij} are approximated as follows with S denoting the fixed setup cost incurring each time an order is placed for any component, $\Psi(z) = \int_{t=z}^{\infty} (t-z) d\Phi(t)$ denoting the standard loss function and $\Phi(\cdot)$ denoting the cumulated distribution function of a standard normal variate:

$$(23) \quad Q_{ij} = \sqrt{\frac{2 \cdot \mu_{ij} \cdot S}{h_{ij}}} \quad \forall (i,j) \in E$$

$$(24) \quad R_{ij} = \tau \cdot \mu_{ij} + \sigma_{ij} \cdot \Psi^{-1} \left(\frac{(1-\beta) \cdot Q_{ij}}{\sigma_{ij}} \right) \quad \forall \quad (i,j) \in E$$

The expected inventory holding cost per time unit HC_{ij} are approximated as sum of two cost components, the inventory holding cost for all safety stocks, HC_{ij}^1, and the inventory holding cost for regular stock, HC_{ij}^2, as follows (for details see Thonemann und Brandeau 2000):

$$(25) \quad HC_{ij}^1 = h_{ij} \cdot \sigma_{ij} \cdot \Psi^{-1} \left(\frac{(1-\beta) \cdot Q_{ij}}{\sigma_{ij}} \right) \quad \forall \quad (i,j) \in E$$

$$(26) \quad HC_{ij}^2 = h_{ij} \cdot \frac{Q_{ij}}{2} \quad \forall (i,j) \in E$$

$$(27) \quad HC_{ij} = HC_{ij}^1 + HC_{ij}^2 \quad \forall (i,j) \in E$$

The expected setup or order cost per time unit is computed by summing up the expected order cost of all components which are derived from dividing the setup cost factor S by the time between orders (fraction of order quantity and expected demand rate):

Table 7: Comparison with simulated annealing of Thonemann and Brandeau (2000)

measure	PRIO	RAND	ANTS
avg gap	6.2	6.8	1.6
max gap	9.2	9.2	4.2
avg cpu	<0.1	6.3	65.1

$$(28) \quad SC_{ij} = \frac{S \cdot \mu_{ij}}{Q_{ij}} \quad \forall (i,j) \in E$$

Depending on their relevance in a component-commonality setting, inventory cost HC_{ij} and setup cost SC_{ij} can be added to other cost components like our fixed and variable cost of CCCP altogether building the cost per component and, thus, arc weights c_{ij} within our solution framework. The additional cost components considered by Thonemann and Brandeau (2000) can also be covered by our solution framework. Their production cost equal the variable cost of the CCCP model and so-called complexity cost can be considered with an extension presented in Section 6.2 for the case of nonlinear increasing fixed cost. Consequently, our solution framework can be applied for the complete commonality problem defined by Thonemann and Brandeau (2000), which is the most general one in existing literature, without difficulty. We tested our solution framework against the simulated annealing approach presented by Thonemann and Brandeau (2000) on their test bed consisting of 20 instances with up to 100 products. The results are summarized in Table 7. With an average gap of merely 1.6 % compared to the average results of the simulated annealing approach specially dedicated to the respective problem our ANTS approach shows competitive. Even PRIO shows satisfying results as an average gap of 6.2 % is achieved in negligible time.

Decreasing variable cost due to learning: An increase of component-commonality entails that remaining components are produced in larger quantity (at least under the premise that commonality does not affect product sales) and, thus, economies of scale can be realized. Although Thonemann and Brandeau (2000) as well as Jans, Degreave, and Schepens (2008) state that learning is an important influencing factor in many real-world

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

commonality problems, it has not been covered by commonality research, thus far. However, learning can be easily incorporated into our solution framework.

For instance, the elementary power model proposed by Wright (1936) assumes the learning curve

(29) $k_n = k_1 \cdot n^{-b}$

where k_x, n and b denote production cost in the x^{th} cycle, number of cycles and learning constant. By building the integral of (29) and rearranging the term total production cost K_n over all n cycles amount to (see Dar-El, 2000, Sec. 3.1.1):

(30) $$K_n = \left(k_1 \cdot n^{(1-b)}\right) \cdot \frac{1}{1-b}$$

With this formula on hand, the total production cost depending on the degree of component-commonality can be calculated and assigned to an arc (i,j) as its arc weight c_{ij}, where v_f^* denotes the maximum value of feature f of all assigned products, see Equations (15):

(31) $$c_{ij} = \left(\left(\sum_{f \in F} k_{fv_f^*} \right) \cdot \left(\sum_{p \in PS_{ij}} d_p \right)^{(1-b)} \right) \cdot \frac{1}{1-b} + K \quad \forall (i,j) \in E$$

Analogously, other learning models (e.g., see Yelle 1979; Dar-El 2000) can be integrated, if they base on (i) initial production cost and (ii) total volume of production, because this information can be readily determined with the help of the data stored with each arc in our solution framework.

6.2 R&D- and engineering-related extensions

Incompatibilities: An important issue during R&D are incompatibilities between certain values of different features. For instance, a subassembly to realize value v of feature f might obstruct the installation slot of another value v' of another feature f'. If these incompatibilities are not considered during the engineering phase, infeasible component specifications might result. Thus, if existent, incompatibilities need to be considered in component-commonality problems. A simple advancement would be to exclude all arcs from the graph, whose assigned product sets lead to component specifications requiring incompatible feature values. However, this is only a heuristic because an upgrade of a subset of values to fix incompatibilities might be less costly than excluding the respective grouping of products. Among all feasible upgrades of values the least costly is to be identified to maintain the property of our solution framework of being able to be applied as an exact approach. If the solution framework is applied with only a subset of product sequences (which is the usual choice for commonality problems of real-world size) and serves as a heuristic, excluding the respective arcs keeps the solution framework simple.

Nonlinear increase of fixed cost: In CCCP fixed cost for component development increase linear in the number of components. This assumption is often not fulfilled in real-world commonality problems (see Thonemann and Brandeau 2000). For instance, some empirical studies reveal an inverted learning curve with an increasing number of components (e.g., see Wildemann 1994, p. 367). To account for arbitrary functions f of fixed cost K depending on the number of components $\mid C \mid$: $K = f(\mid C \mid)$, a special shortest path procedure needs to be applied. This procedure is an adoption of the approach of Saigal (1968), which determines the shortest among all paths with a given number of k arcs. In our modified approach, first, all shortest paths with k arcs for any possible arc number $k = 1, \ldots, \mid P \mid$ are determined, where only variable cost are considered as arc weights. Then, fixed cost $f(k)$ for k components are added to any of $\mid P \mid$ shortest paths determined and the minimum over these solutions is the overall optimal solution for a given set of sequences. Overall runtime complexity of our modified approach is $O(\mid P \mid \cdot \mid V \mid^2)$ with $\mid V \mid$ denoting the number of nodes in the grouping graph. The additional notation required is summarized in Table 8. A formal description is as follows:

1. Determine the structure of the grouping graph, where fixed costs are excluded from calculating arc weights. If nonlinear fixed costs are the only alteration compared to base model CCCP arc weights equal variable cost VC_{ij} (see (15)): $c_{ij} = VC_{ij} \ \forall (i,j) \in E$.
2. Initialize the following data: $R_j(1) = c_{0j} \ \forall j \in$

BuR – Business Research
Official Open Access Journal of VHB
Verband der Hochschullehrer für Betriebswirtschaft e.V.
Volume 2 | Issue 1 | May 2009 | 86–106

Table 8: Additional notation for nonlinear fixed cost

k	number of arcs applied in a path between two nodes
V^+	Set of nodes without start node 0: $V^+ = V \setminus \{0\}$
$R_j(k)$	length of the shortest path to node j among all paths with k arcs
$P_j(k)$	ordered sequence of nodes on the shortest path to node j among all paths with k arcs

V^+; $P_j(1) = \langle 0, j] \, \forall j \in V^+$; k:=1.

3. Recursively determine length and nodes on the shortest path to any node $j \in V^+$ where the number of arcs is restricted to exactly k: $R_j(k+1) = \min\{R_i(k) + c_{ij} \mid (i,j) \in E\} \, \forall j \in V^+$ and $P_j(k+1) = \langle P_{i^*}(k), j] \, \forall j \in V^+$, where i^* is the respective predecessor node on the shortest path and $\langle P, j]$ denotes that element j is appended to list P.

4. Set $k := k+1$ and goto Step (3), unless $k >\mid P \mid$.

5. Add nonlinear fixed cost (represented by function $f(k)$) to any shortest path. The minimum cost over all k solutions is the overall cost minimum for the given set of sequences: $D^* = \min\{R_{|V^+|}(k) + f(k) \mid k = 1, \ldots, \mid P \mid\}$.

6.3 Marketing related extensions

Finally, component-commonality also has an important influence on the market side (see Sections 1 and 2). Although some analytical papers on the relationship between component-commonality and sales have been published in the recent years, the paper of Jans, Degreave, and Schepens (2008) was the first to integrate sales aspects in an optimization model on commonality. In their industrial case study, a given set of products (power tools) needs to be partitioned in families. Within each family all products receive a common stage and frame, where the one product per family having the largest engine determines the requirements for both components (downward substitutability). As hitherto, in such a setting component-commonality influences the variable cost of the common components (stage and frame) for each family, whereas additional groups entail fixed cost for component development. Additionally, prices for the power tools are calculated on a cost-plus basis, so that unit costs are simply increased by a given percentage mark-up. Then, sales are anticipated with the help of a given price elasticity when compared to the old selling price, so that the net present value of resulting returns can be maximized. As percentage mark-up, price elasticity and old selling price per product are all given parameters the solution of the Jans, Degreave, and Schepens (2008) problem only depends on the grouping of products and can, thus, be easily solved with our solution framework. We tested our framework with the case study presented by Jans, Degreave, and Schepens (2008), which consists of 8 power tools (products) for which engines (components) have to be determined. We could solve the problem with a complete enumeration within 6.2 CPU-seconds, which is considerably faster than the off-the-shelf solver utilized by Jans, Degreave, and Schepens (2008) (52 CPU-seconds).

7 Conclusion

The paper on hand introduces a two-stage graph approach, which is flexible enough to be applied to a wide range of component-commonality problems. The solution performance of the procedure was shown to be very promising if applied to a basic component-commonality problem. Whether similar results can be obtained for a broader class of component-commonality problems cannot be answered at present due to the apparent lack of benchmark problems and comparable procedures. As the shortest path problems in the second stage are always solved to optimality (except for incompatibilities, see Section 6.2), irrespective of the considered extensions, the ability of identifying promising product sequences will most likely have the strongest impact on the solution quality. Our computational study revealed that our solution framework shows robust solution quality irrespective of the first-stage procedure, so that it can be expected that this will hold for the vast majority of presented extensions alike. However, it remains up to future research to further support this conjecture.
Another promising field for future commonality research would be to consider the interrelationship between different components. On the one hand, a technical interrelationship might be rel-

evant, whenever, for instance, commonality leads to some heavier multi-purpose components, which altogether would exceed a given maximum weight allowed for the final product or increase its energy demand. On the other hand, component-commonality leads to a blending of products in the customer's perception, which, however, can be compensated with other exceptional properties (components) of the respective product. Thus, all types of components and decisions on their levels of commonality are interrelated with regard to the customer's utility valuation of the products. To explicitly cover this effect, the advancement of relating component-commonality via a cost-plus price setting and a price elasticity (see Jans, Degreave, and Schepens 2008, and Section 6.3) is not sufficient. Consequently, joint optimization models of product line selection (e.g., see Green and Krieger 1985; Nair, Thakur, and Wen 1995) and component-commonality are required to capture the overall decision problem in a more detailed fashion.

Acknowledgements

We want to thank the department editor and two anonymous referees for very valuable suggestions which helped to improve the paper considerably.

References

Avella, Pasquale, Maurizio Boccia, Carmine Di Martino, Giuseppe Oliviero, Antonio Sforza, and Igor Vasil'ev (2005): A decomposition approach for a very large scale optimal diversity management problem, *4OR*, 3 (1): 23-37.

Baker, Kenneth R. (1985): Safety stocks and component commonality, *Journal of Operations Management*, 6 (1): 13-22.

Baker, Kenneth R., Michael J. Magazine, and Henry L. W. Nuttle (1986): The effect of commonality on safety stock in a simple inventory model, *Management Science*, 32 (8): 982-988.

Boysen, Nils and Malte Fliedner (2008): A versatile algorithm for assembly line balancing, *European Journal of Operational Research* , 184 (1): 39-56.

Boysen, Nils, Malte Fliedner, and Armin Scholl (2009a): Level scheduling of mixed-model assembly lines under storage constraints, *International Journal of Production Research*, 47 (1): 2669-2684.

Boysen, Nils, Malte Fliedner, and Armin Scholl (2009b): Assembly line balancing: Joint precedence graphs under high product variety, *IIE Transactions*, 41 (1): 183-193.

Briant, Olivier (2000): *Etude théorique et numérique du problème de la gestion de la diversité*, Ph.D. thesis, Institut national polytechnique de Grenoble, Grenoble, France.

Briant, Olivier and Denis Naddef (2004): The optimal diversity management problem, *Operations Research*, 52 (4): 515-526.

Burbidge, John L. (1991): Production flow analysis for planning group technology, *Journal of Operations Management*, 10 (1): 5-27.

Collier, David A. (1982): Aggregate safety stock levels and component part commonality, *Management Science*, 28 (11): 1296-1303.

Dar-El, Ezey M. (2000): *Human learning: From learning curves to learning organizations*, 1st ed., Springer: Berlin.

Desai, Preyas, Sunder Kekre, Suresh Radhakrishnan, and Kannan Srinivasan (2001): Product differentiation and commonality in design: Balancing revenue and cost drivers, *Management Science*, 47 (1): 37-51.

Dogramaci, Ali (1979): Design of common components considering implications of inventory costs and forecasting, *AIIE Transactions*, 11 (2): 129-135.

Dorigo, Marco, Gianni Di Caro, and Luca M. Gambardella (1999): Ant algorithms for discrete optimization, *Artificial Life*, 5 (2): 137-172.

Ervolina, Thomas R., Markus Ettl, Young M. Lee, and Daniel J. Peters (2009): Managing product availability in an assemble-to-order supply chain with multiple customer segments, *OR Spectrum*, 31 (1): 257-280.

Eynan, Amit and Meir J. Rosenblatt (1996): Component-commonality effects on inventory costs, *IIE Transactions*, 28 (2): 93-104.

Fisher, Marshall, Kamalini Ramdas, and Karl Ulrich (1999): Component sharing in the management of product variety: A study of automotive braking systems, *Management Science*, 45 (3): 297-315.

Gerchak, Yigal, Michael J. Magazine, and Bruce A. Gamble (1988): Component commonality with service level requirements, *Management Science*, 34 (6): 753-760.

Green, Paul E. and Abba M. Krieger (1985): Models and heuristics for product line selection, *Marketing Science*, 4 (1): 1-19.

Gupta, Saurabh and Vish Krishnan (1999): Integrated component and supplier selection for a product family, *Production and Operations Management*, 8 (2): 163-182.

Hanssmann, Friedrich (1957): Determination of optimal capacities of service for facilities with a linear measure of ineffciency, *Operations Research*, 5 (5): 713-717.

Hesse, Hans Sebastian and Jayashankar M. Swaminathan (2006): Product line design with component commonality and cost-reduction effort, *Manufacturing Service Operations Management*, 8 (2): 206-219.

Hillier, Mark S. (2000): Component-commonality in multiple-period, assemble-to-order systems, *IIE Transactions*, 32: 755-766.

Hillier, Mark S. (2002): Using commonality as backup safety stock, *European Journal of Operational Research*, 136 (2): 353-365.

Jans, Raf, Zeger Degreave, and Luc Schepens (2008): Analysis of an industrial component commonality problem, *European Journal of Operational Research*, 186 (2): 801-811.

Karup, Jakob and Peter M. Pruzan (1983): The simple plant location problem: Survey and synthesis, *European Journal of Operational Research*, 12 (1): 36-81.

Kim, Kilsun and Dilip Chhajed (2000): Commonality in product design: Cost savings, valuation change and cannibalization, *European Journal of Operational Research*, 125 (3): 602-621.

King, John R. and Vias Nakornchai (1982): Machine-component group formation in group technology: Review and extension, *International Journal of Production Research*, 20 (2): 117-133.

Klose, Andreas and Andreas Drexl (2005): Facility location models for distribution system design, *European Journal of Operational Research*, 162 (1): 4-29.

Krishnan, Viswanathan, Rahul Singh, and Devanath Tirupati (1999): A model-based approach for planning and developing a family of technology based products, *Manufacturing Service Operations Management*, 1 (2): 132-156.

Labro, Eva (2004): The cost effects of component commonality: a literature review through a management-accounting lens, *Manufacturing Service Operations Management*, 6 (4): 358-367.

Lang, Jan C. and Wolfgang Domschke (2008): Efficient reformulations for dynamic lot-sizing problems with product substitution, *OR Spectrum*, 23 (1): 1-29.

Ma, Shihua, Weii Wang, and Liming Liu (2002): Commonality and postponement in multistage assembly systems, *European Journal of Operational Research*, 142 (3): 523-538.

McClain, John O., William L. Maxwell, John A. Muckstadt, L. Joseph Thomas, and Elliott N. Weiss (1984): Comment on: Aggregate safety stock levels and component part commonality, *Management Science*, 30 (6): 772-773.

Nair, Suresh K., Lakshman S. Thakur, and Kuang-Wei Wen (1995): Near optimal solutions for product line design and selection: Beam search heuristics, *Management Science*, 41 (5): 767-785.

Pentico, David W. (2008): The assortment problem: A survey, *European Journal of Operational Research*, 190 (2): 95-309.

Perera, H. S. C., Nagen Nagarur, and Mario T. Tabucanon (1999): Component part standardization: A way to reduce the life-cycle costs of products, *International Journal of Production Economics*, 60-61 (1): 109-116.

Ramdas, Kamalini (2003): Managing product variety: An integrative review and research directions, *Production and Operations Management*, 12 (1): 79-101.

Ramdas, Kamalini and Mohanbir S. Sawhney (2001): A Cross-functional approach to evaluating multiple line extensions for assembled products, *Management Science*, 47 (1): 22-36.

Ramdas, Kamalini, Marshall Fisher, and Karl Ulrich (2003): Managing variety for assembled products: Modeling component systems sharing, *Manufacturing Service Operations Management*, 5 (2): 142-156.

Rutenberg, David P. (1971): Design commonality to reduce multi-item inventory: Optimal depth of a product line, *Operations Research*, 19 (2): 491-509.

Sadowski, Wieslaw (1959): A few remarks on the assortment problem, *Management Science*, 6 (1): 13-24.

Saigal, Romesh (1968): A constrained shortest route problem, *Operations Research*, 16 (1): 205-209.

Stützle, Thomas and Marco Dorigo (1999): ACO algorithms for the quadratic assignment problem, in: David W. Corne, Marco Dorigo, and Fred Glover (eds.): *New Ideas in Optimization*, McGraw-Hill: 33-50.

Swaminathan, Jayashankar M. (2001): Enabling customization using standardized operations, *California Management Review*, 43 (3): 125-135.

Swaminathan, Jayashankar M. and Hau L. Lee (2004): in: S.C. Graves, and A.G. De Kok (eds.): *Handbooks in Operations Research and Management Science 11, Supply Chain Management: Design, Coordination, and Operation*, Elsevier: 199-226.

Swift, Ken G., Julian Booker, and Nigel Foden Edmondson (2004): Strategies and case studies in assembly system selection, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 218 (7): 675-688.

Tallon, William J. (1989): A comparative analysis of master production scheduling techniques for assembly-to-order products, *Decision Science*, 20 (3): 492-506.

Thomas, Lawrence Dale (1991): Commonality analysis using clustering methods, *Operations Research*, 39 (4): 677-680.

Thonemann, Ulrich W. and Margaret L. Brandeau (2000): Optimal commonality in component design, *Operations Research*, 48 (1): 1-19.

Ulrich, Karl (1995): The role of product architecture in the manufacturing firm, *Research Policy*, 24 (3): 419-440.

Wagner, Harvey M. and Thomson M. Whitin (1958): Dynamic version of the economic lot size model, *Management Science*, 5 (1): 89-96.

Wildemann, Horst (1994): *Fertigungsstrategien,* 2nd ed., TCW: München.

Wright, T.P. (1936): Factors Affecting the Cost of Airplanes, *Journal of Aeronautical Sciences*, 3 (2): 122-128.

Yelle, Louis E. (1979): The learning curve: Historical review and comprehensive survey, *Decision Science*, 10 (2): 302-328.

Biographies

Nils Boysen is Professor of Operations Management at the Friedrich-Schiller-University of Jena (Germany). He received a Diploma Degree and a PhD in Business Administration from the University of Hamburg. From 2000-2002 he worked for IBM Global Services. His research interests lie in production and operations management as well as optimization techniques. His work has been published in many international journals including European Journal of Operational Research, OR Spectrum and IIE Transactions.

Armin Scholl has held the Chair of Decision Analysis and Business Administration at the Friedrich-Schiller-University Jena (Germany) since 2000. He received a Diploma Degree in Economics and Computer Science and a PhD in Business Administration from Darmstadt University of Technology. His research interests are combinatorial optimization, preference measurement, multi-attribute decision making, planning systems, distributed planning and heuristic decision making. He has published many articles in international journals including European Journal of Operational Research, INFORMS Journal on Computing, International Journal of Production Research.